Atualização do robô recepcionista utilizando robótica em nuvem

Bolsista Marcos Vinicius Cruz (CTI) mvcruz@cti.gov.br

Resumo

Robôs Socialmente interativos devem interagir com humanos de forma apropriada respeitando regras sociais, para executar essa tarefa é requerido que o robô tenha um bom orquestrador de tarefas para administrar suas diversas funcionalidades para executar uma interação socialmente aceitável. Neste trabalho é apresentado a nova versão da recepcionista utilizando uma arquitetura híbrida conhecida como HBBA (Hybrid Behavior Based Architecture) que é apta para o uso de robótica em nuvem e novos módulos do sistema que são vistos como serviços na arquitetura.

Palavras-chave: Interação Humano-Robô, Robótica em nuvem, Robôs Socialmente Interativos, Arquitetura Robótica.

1. Introdução

No contexto do projeto Fapesp ROSANA, que possui um elevado grau de percepção, fruto de uma arquitetura distribuída de GPUs para reconhecimento de objetos por meio de Redes Neurais Convolucionais Regionais (PAIVA, 2020), motivou a adaptação robô recepcionista para uma plataforma robótica móvel com o intuito de realizar tarefas como aproximação e interação com usuários, utilizando serviços disponibilizados via rede, visando minimizar o consumo de processador embarcado no robô, pois fica inviável alocar todos recursos que requerem alto consumo de processamento, consequentemente, pode gerar sobrecarga.

Neste sentido realizou-se estudos conceituais para a incorporação de uma arquitetura de robótica híbrida HBBA e a migração de modelo baseado em orientação de objetos para uma baseada em serviços incorporando a distribuição do sistema através do middleware ROS.

2. Robótica em nuvem

Robótica em nuvem é definida como qualquer sistema robótico autônomo que depende de dados ou serviços que são acessados pela rede para realizarem suas operações. Neste tipo de sistemas, sensores, processamento e memória são distribuídos, não sendo alocados em um único sistema (KUFFNER, 2010). Há muitas vantagens na robótica em nuvem como: acesso a dados atualizados, paralelismo no processamento de dados de acordo com a demanda evitando sobrecarga do sistema, compartilhamento de informações captadas por sensores, entre outras.

Segundo Chen et. al., é possível enumerar 4 tipos de modelos para robótica em nuvem (2010): Infra-estrutura como um serviço (IaaS); Plataforma como um serviço (PaaS); Software como um serviço (SaaS); Robô como serviço (RaaS).

Os modelos referenciados necessitam de uma arquitetura orientada à serviço estendida para o conceito de robótica em nuvem, onde existem diferentes camadas de abstração que se comunicam através de algum protocolo de comunicação pré-definido. A performance de um sistema em nuvem está diretamente relacionado à velocidade de transmissão da rede, mas o desempenho do sistema pode gradativamente diminuir se não existir um ambiente estruturado, sem padrões de comunicações definidos e uma arquitetura que não atenda as demandas exigidas para robótica em nuvem (SAHA, 2018). Para o trabalho apresentado foi definido como protocolo de comunicação *publish/subscribe* e a arquitetura utilizada HBBA, onde ambos serão apresentados nos próximos capítulos.

3. Padrão de Comunicação Publish/Subscribe

As principais entidades em um sistema que utiliza o padrão *publish/subscribe* para comunicação são os *publishers* e *subscribers* de dados. O *publisher* detecta um evento e então publica na forma de notificação. A notificação faz o encapsulamento do conteúdo relacionado ao evento observado. A notificação também pode ser chamada de mensagem de evento (BACON, 2008). Após um evento ser publicado, o sistema *publish/ subscribe* deve entregar a mensagem para a todos os módulos e serviços que tem interesse na informação, através do *subscriber*. O *subscriber* é um elemento que expressa interesse em determinados eventos que ocorrem no sistema (MILLARD, 2010).

Um evento representa qualquer estado de transição que tenha ocorrido em um módulo ou serviço, como um estado de login bem sucedido ou, quando um alarme de incêndio é ativado. Eventos podem ser categorizados pelos seus atributos, podendo ser: caracteŕisticas físicas; informações espaciais ou temporais; taxonomia; mensagem de avisos; entre outros (BACON, 2008).

A solução usada no sistema da recepcionista é feita pelo ROS onde as mensagens são transmitidas para todo o sistema e todos os serviços enxergam essas mensagens, entretanto apenas será processada por um serviço se existir um *subscriber* ativo para aquele evento em questão.

4. Hybrid Behavior Based Architecture

A arquitetura *Hybrid Behavior-Based Architecture* fornece um conjunto de elementos para controlar o sistema, seguindo um padrão para auxiliar a modelagem de um orquestrador de tarefas. O controle baseado em comportamentos possui as mesmas características de uma arquitetura híbrida, agindo de forma reativa (com tomadas de reações rápidas) e deliberativa (que envolve planejamento em suas decisões) definidas a partir de camadas. Entretanto não existe uma camada intermediária entre elas, os componentes possuem uma representação de escala de tempo uniforme (FERLAND,2017).

Os comportamentos são processos concorrentes que podem receber informações de sensores ou outros comportamentos e, após processá-los envia os resultados para os atuadores do robô ou comportamentos para atingir algum objetivo. A HBBA é composta por 3 camadas: *organization layer* onde estão as motivações; *coordination layer* contendo o interaction workspace e seus módulos associados; *behavioral layer,* contendo os módulos de percepção e comportamento (FERLAND, 2017).

Motivações são módulos independentes de alto nível que gerenciam metas para o robô. Metas são descritas como desejos que podem inibir ou incentivar ações, como falar ou andar, mas também podem prover informações como reconhecimento de objetos. Módulos de percepção pegam as informações brutas de sensores e transformam em *percepts* para os comportamentos. Como detecção do rosto, onde imagens brutas da câmera são processadas para produzir a localização do rosto de uma pessoa presente. Comportamento são processos independentes que processam os *percepts* e geram comandos arbitrando eles baseado em um esquema de prioridades.

O *Intention Workspace* é responsável por processar todos os desejos do robô, que são previamente modelados pelo programador, e converte o conjunto de desejos em intenções (*intent*) para o robô. Eles descrevem o que o robô deve fazer. A transcrição de desejo para intenções é feito pelo IW translator que possui um banco de dados de estratégias para realizar a conversão. Os desejos são categorizados por classes e cada estratégia descreve uma maneira para cumprir uma única classe de desejos e as intenções do robô são enviadas de acordo com o fluxo de dados entre os módulos da arquitetura por meio de filtros. É importante ressaltar que apenas um desejo por classe pode ser realizado ao mesmo tempo para evitar conflitos, os desejos são parametrizados por intensidades, e a prioridade é dada aos que têm maior intensidade (FERLAND,2017).

As estratégias são caracterizadas pela utilidade e custos em recursos. Uma estratégia com utilidade superior geralmente requer mais recursos e vice-versa. O papel do *IW Translator* deve cumprir tantos desejos quanto possível, respeitando os recursos do robô.

5. Caracterização do Sistema

O sistema desenvolvido possui uma série de ferramentas relacionadas a processamento de imagem, reconhecimento de voz e comunicação, necessárias para seu adequado funcionamento.

Como forma de gatilho (trigger) que habilita a interação dos módulos, utilizou-se do reconhecimento facial, ou seja, o robô passa a interagir com o usuário assim que ele identifica um rosto à sua frente. A captura de imagem e reconhecimento facial foram feitas em cima da biblioteca OpenCV, como mostra no documento técnico registrado no sigtec de código TAC0002-2020.

Para o trabalho é necessária a utilização de um motor de conversão de áudio para texto eficiente. Assim foi decidido pela escolha do API *Google Speech-to -Text*, pois este oferecia uma documentação clara e recursos suficientes para os objetivos almejados, além de boa confiabilidade em seus resultados. (SHAKHOVSKA, 2019)

Além da transcrição de áudio, para uma melhor análise do contexto da fala, foi utilizada a biblioteca Spacy. Esta é uma biblioteca de código aberto de processamento de linguagem natural que suporta uma variedade de tarefas, incluindo marcação, reconhecimento de entidade nomeada, análise de dependência, etc. (PARTALIDOU, 2019) É através dela que é realizada a análise morfológica do texto, onde a marcação gramatical, indica os nomes e termos a serem buscados no banco de dados.

6. Distribuição Horizontal do Sistema

O sistema foi implementado segundo o paradigma de arquitetura orientada a serviços (SOA). Este tipo de design de software se utiliza de componentes (serviços) reutilizáveis através de uma interface de serviços em rede com uma linguagem de comunicação comum. Segundo (PAPAZOGLOU, 2013), o SOA é uma forma de reorganizar um conjunto de aplicativos de software previamente isolados e infraestrutura de suporte, em um conjunto interconectado de serviços, cada um acessível por meio de interfaces padrão e protocolos de mensagens. Assim, devido a distribuição dos serviços, o sistema se caracteriza como sendo de escalonamento horizontal, onde seu processamento é dividido em diferentes máquinas.

Para a adaptação de módulos para um serviço leva em consideração 3 categorias principais. Primeiramente a camada de abstração de hardware (HAL), que consiste, neste caso, da câmera, do microfone e dos módulos de controle. Em seguida os módulos principais, onde é definido a ordem de execução das tarefas. E por último os módulos de motivação, percepção e comportamento de nível superior, onde serão definidos a ordem de execução.

Os módulos são ativados e desativados a partir de arquivos de configuração que definem a ordem e em qual momento eles devem ser executados. Para isso, é necessária a configuração de dois metafiles. Um são launcher que definem o conjunto de módulos que serão utilizados em um serviço para o HBBA saber como acessá-los, o segundo são os arquivos de configuração que padronizam os acessos para o formato do HBBA, nele são definidos os comportamentos, estratégias e desejos.

Com o intuito de tentar melhorar o tempo e acurácia das respostas do robô recepcionista, investigamos o uso de alguns modelos de redes neurais e seus impacto para o tratamento de perguntas do usuário, sendo estes transformers e Sequence-to-Sequence model: uma arquitetura de redes neurais profundas (BRAŞOVEANU,2020) e uma classe especial de redes neurais recorrente (SUTSKEVER, 2014), respectivamente.

8. Contribuições em outros projetos

Grupo de Ciência de Dados: Participação no grupo de dados como consultor para auxiliar o grupo com conhecimento na área e ajuda na análise de dados análise de dados de energia elétrica para análise e predição com o intuito de eficiência energética predial.

Navegação Socialmente aceitável: Auxílio na aquisição da orientação e posição de pessoas no espaço 3D utilizando kinect.

Detecção de emoções complexas: Acompanhamento periódico no projeto de detecção de emoções complexas para validação do trabalho e discussões em grupo.

9. Prova de Conceito com HBBA e sistema distribuído

Para realizar os experimentos partimos das premissas que modelo seq2seq irá demorar mais tempo para responder as requisições devido ao gargalo de conexão da rede interna e limitações de hardware que afetam diretamente a performance da rede neural, enquanto esperamos que o HBBA utilize menos recursos de CPU e memória devido sua administração de recursos.

Experimentos são realizados em dois modelos de máquinas diferentes onde que a versão original do robô recepcionista e o avatar são executados em uma máquina windows 10 com 8 GB de memória e um processador Intel(R) HD Graphics 4600 e os serviços são executados em uma máquina com SO linux 18.04, 8 GB de memória e um processador Radeon HD 8490. Abaixo segue a descrição dos experimentos realizados:

a. Desempenho a médio prazo: consistiu de uma análise de desempenho a médio prazo do consumo de CPU e memória das máquinas. Neste experimento, ambos os modelos ficaram em execução contínua durante o período de 5 dias, onde foram efetuados registros quanto ao desempenho de CPU e consumo de memória.

b. Performance em interações: Neste experimento é avaliado o tempo de respostas e a precisão das respostas entre a versão original e a nova da recepcionista, foram feitas 2 perguntas para ambos os agentes: Saudação e requisição de uma informação interna com o nome interno da base de conhecimento.

Os gráficos abaixo apresentam desempenho de CPU e memória (Figura 1), onde o uso de CPU é representado em porcentagem e a memória e *bytes*. E a Figura 2 apresenta os dados relacionados ao experimento do tempo de respostas para cada pergunta.

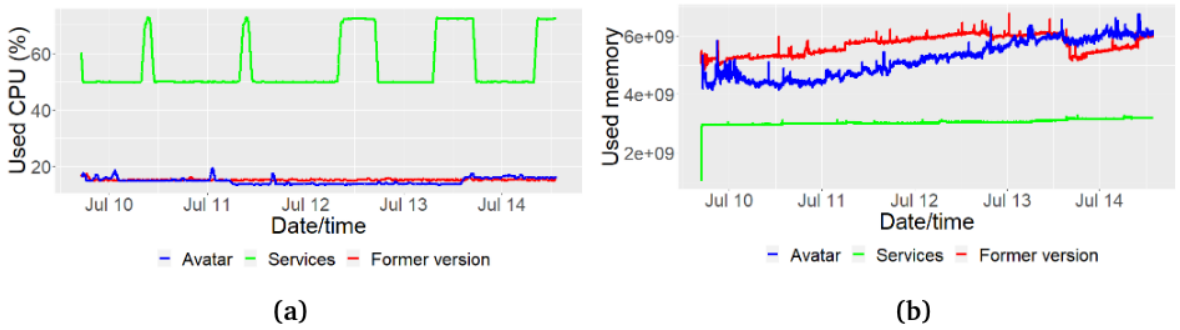

Figura 1 – (a) uso de CPU a cada 15 minutos. (b) consumo de memória a cada 15 minutos.

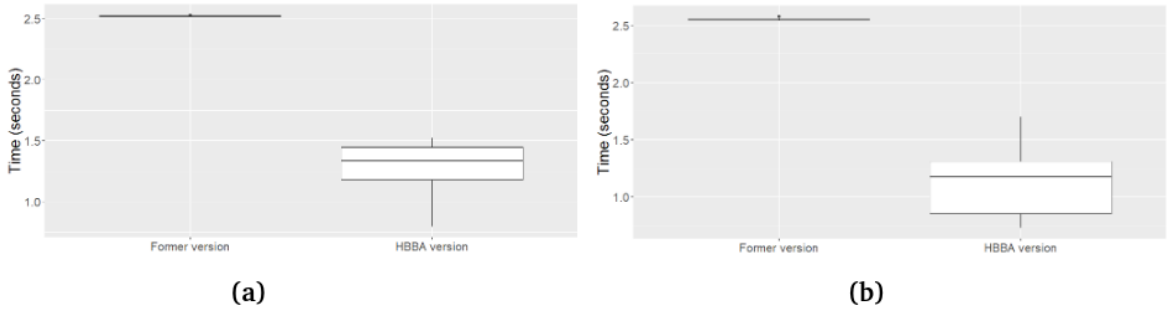

Figura 2 – (a) tempo de resposta para saudação. (b) tempo de resposta para busca de pessoa.

10. Conclusão

O trabalho objetivou apresentar as atividades do bolsista durante o ano de 2021. Foi realizada uma adaptação da arquitetura robótica HBBA para um robô recepcionista. Os resultados sugerem que o consumo de CPU em médio prazo foi constante e de menor intervalo de

resposta em comparação ao sistema anterior. Em relação ao uso de memória, o HBBA a reserva enquanto cria variáveis, alocando recursos de acordo com o grau de intensidade para reduzir a fragmentação da memória.

Para trabalhos futuros, pretende-se utilizar-se de tal robô recepcionista em um robô móvel, permitindo a exploração de tarefas relacionadas, também, à navegação social do robô.

Referências

BACON, Jean et al. Access control in publish/subscribe systems. In: Proceedings of the second international conference on Distributed event-based systems. 2008. p. 23-34.

BRAŞOVEANU, Adrian MP; ANDONIE, Răzvan. Visualizing transformers for nlp: a brief survey. In: 2020 24th International Conference Information Visualisation (IV). IEEE, 2020. p. 270-279.

CHEN, Yinong; DU, Zhihui; GARCIA-ACOSTA, Marcos. Robot as a service in cloud computing. In: 2010 Fifth IEEE International Symposium on Service Oriented System Engineering. IEEE, 2010. p. 151-158.

FERLAND, François et al. Coordination mechanism for integrated design of human-robot interaction scenarios. Paladyn, Journal of Behavioral Robotics, v. 8, n. 1, p. 100-111, 2017

SUTSKEVER, Ilya; VINYALS, Oriol; LE, Quoc V. Sequence to sequence learning with neural networks. In: Advances in neural information processing systems. 2014. p. 3104-3112.

KUFFNER, James. Cloud-enabled humanoid robots. In: Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on, Nashville TN, United States, Dec. 2010.

MILLARD, Peter; SAINT-ANDRE, Peter; MEIJER, Ralph. XEP-0060: publish-subscribe. XMPP Standards Foundation, v. 1, p. 13, 2010.

PAIVA, Pedro VV et al. ROSANA: Robot for Social Interaction in Unstructured Dynamic Environments. In: 2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE). IEEE, 2020. p. 1-6.

PAPAZOGLOU, Mike P. Service-oriented computing: Concepts, characteristics and directions. In: Proceedings of the Fourth International Conference on Web Information Systems Engineering, 2003. WISE 2003. IEEE, 2003. p. 3-12

PARTALIDOU, Eleni et al. Design and implementation of an open source Greek POS Tagger and Entity Recognizer using spaCy. In: 2019 IEEE/WIC/ACM International Conference on Web Intelligence (WI). IEEE, 2019. p. 337-341.

SAHA, Olimpiya; DASGUPTA, Prithviraj. A comprehensive survey of recent trends in cloud robotics architectures and applications. Robotics, v. 7, n. 3, p. 47, 2018.

SHAKHOVSKA, Nataliya; BASYSTIUK, Oleh; SHAKHOVSKA, Khrystyna. Development of the Speech-to-Text Chatbot Interface Based on Google API. In: MoMLeT. 2019. p. 212-221.